11/7/2018

XV Convegno Italiano degli Utenti di Stata Bologna, 15-16 November, 2018

## **Calling External Routines in Stata**

Giovanni Cerulli and Antonio Zinilli

**IRCrES-CNR** 



#### Motivation

Stata allows to call **external routines**, written in other software, to perform specific tasks within Stata

This talk offers some insights on how to develop a **Stata ADO file** embedding an external software routine (R, in this case)

□We provide a user-written Stata module stree, written to allow users to run regression trees (a Machine Learning technique currently unavailable in Stata) by calling back the R software

#### Three "**R** ==> Stata" alternatives

#### Rcall

Integrating R with Stata by allowing inter-process Very flexible, but a bit timecommunication between the two software (Haghish, consuming to learn E.F., 2017) Rsource Very easy to use, but not For running an R source program from an inline really handy for ADO files sequence of lines or from a file, in batch mode from within Stata shell More general approach, Allowing to send commands to your operating system apparently more complicated, or to enter your operating system for interactive use but finally easy to use 3

# The Basics of Decision Trees

Decision trees can be applied to both regression and classification problems

#### Example of a Decision Tree



- For the Hitters data, a regression tree for predicting the log salary of a baseball player, based on the number of years that he has played in the major leagues and the number of hits that he made in the previous year.
- At a given internal node, the label (of the form  $X_j < t_k$ ) indicates the left-hand branch emanating from that split, and the right-hand branch corresponds to  $X_j \ge t_k$ . For instance, the split at the top of the tree results in two large branches. The left-hand branch corresponds to Years>=4.5, and the right-hand branch corresponds to Years>=4.5.
- The tree has two internal nodes and three terminal nodes, or leaves. The number in each leaf is the mean of the response for the observations that fall there.

## Interpretation of Results

Years is the most important factor in determining Salary, and players with less experience earn lower salaries than more experienced players.

Given that a player is less experienced, the number of Hits that he made in the previous year seems to play little role in his Salary.

But among players who have been in the major leagues for five or more years, the number of Hits made in the previous year does affect Salary, and players who made more Hits last year tend to have higher salaries.



5

Finding the optimal number of terminal nodes: Optimal-Tree detection

As other Machine Learning methods facing a bias-variance trade-off, the **optimal tree** is the one "balancing" *bias reduction* and *variance increase*, within the *largest* possible tree  $T_0$  obtained from the *training* dataset.

The problem can be solved via a **penalization** approach, which penalizes too complex trees by at the same time allowing a not too large bias

This can be done via **optimal tree-pruning** 

Example - Regression tree for the Hitters data 1





### Example - Regression tree for the Hitters data 2



MSE for the **training**, the **cross-validation**, and the **test** as a function of the **number of terminal nodes** in the pruned tree.

### Example - Regression tree for the Hitters data 3



10



### A Stata/R user-written ADO-file template

- 1. Write **srprog.ado**, a master Stata program calling back Stata subprograms containing R code
- 2. Write **srprog1.ado**, **srprog2.ado**,... the needed Stata subprograms containing R code and generating an R program called **srprog.R**
- 3. Write the Stata program **runR.ado** executing **srprog.R** via the **shell** Stata command

11

#### A Stata/R user-written ADO-file template – step 1

Write a Stata program called srprog	
Set the main directory as the present working directory (pwd)	<pre>************************************</pre>
Export the ".dta" dataset in the current memory into a ".csv" called "mydata.csv"	<pre>qui export delimited `var' using "mydata.csv", nolabel replace ************************************</pre>
Run a program srprog1 containing an R script conditionally on option1	runR , options // Stata program running R ************************************
Execute the Stata command <b>runR</b> to make Stata able to let R to do its job.	

#### A Stata/R user-written ADO-file template – step 2



#### A Stata/R user-written ADO-file template - step 3





#### The Stata user-written command stree

#### Options

model(modeltype) specifies the type of model, where:

modeltype	
tree	Fits a tree, either unpruned, pruned, and optimal via CV
tree_rf	Fits a tree using random forests
tree_bag	Fits a tree using bagging
tree_boost	Fits a tree using the boosting algorithm

- **op\_sys**(ostype) specifies the operating system you are working with. Two options for ostype are available, "WIN" (Windows) and "IOS" (MAC)
- prune(integer) specifies the optimal pruned tree at size (number of nodes) "integer"; for instance prune(5), prune(8), ...

#### Application to a classification tree (using sctree\*)



\* For fitting a regression tree the companion command is called srtree



### R output visible as Stata output - 1

Ands here  ands here      ands here      _rc      Users\Ceruli\Dr  s\Ceruli\Dropb  s\Ceruli\Dropb	> Beginning of R output from source file: mytree.R R version 3.5.1 (2018-07-02) "Teather Spray" Cogyright (C) 2018 The R Toundation for Statistical Computing Flatform: 28-6_4-4-ang027_24(46-bit)	◆ Variables ◆ Filter Name
siCerulli/Droph	<pre>b D m setTemme likero ed D riasenisto SETEA ALCEMA GAMMILA. Sizam ben litel is eptots relativibulic, as as costo certe condision. Servivi 'license()' o 'licence()' per dettagli su come distribuirlo. A D um propetto di collaborazione con soli contributi esterni. Servivi (demotivers()' per angiori informazione i e'itation()' per aspere come oitare R o i pacchetti di R malle pubblicazioni. Servivi (demotivers()' per angiori informazione e'itation()' per aspere come oitare R o i pacchetti di R malle pubblicazioni. Servivi (demotivers()' per ma dimostrazione, 'hab()') per la guida in lines, o 'hab,-sfart()' per l'hab; marchy cobile con browser TREL. Servivi 'q0' per ascire da R. ) list.or packanges &lt;- (riconign''tree'', "MASS"."ISLR') &gt; new poskages &lt;- (itoring''tree', "MASS"."ISLR') &gt; new poskages &lt;- (itoring''tree', "MASS"."ISLR') &gt; interpriveres poskages) install.packages(sew.packages intstalled.packages()[,"Fackage")]) &gt; lifcarpt(tree; packages) &gt; secture() &gt; secture() &gt; secture() &gt; secture() &gt; secture() &gt; secture() &gt; secture() &gt; d Load : &gt; d Load : &gt; libearry("tree") &gt; libearry("tree") &gt; libearry("tree") &gt; libearry("tree") &gt; libearry("tree") &gt; libearry("tree") &gt; libearry("tree") &gt; libearry("tree") &gt; libearry(Stere") &gt; libearry(Stere) &gt; libearry(St</pre>	Sales CompP Income Adverti Popula Price Age Educati High US Urban Shelvel
	> #summary(tree.data) > #stattinee.data,min(") > #stattinee.data,pentive) > #fortthee.data,pentive) > #forvitheermodiataise data > prome.dataeprome.tree(tree.data,best=6) > summary(prome.data)	E.
		*

## **R** output visible as Stata output - 2

<pre>classification tree: imp:tree(tree = tree.dt.g.nodes = c(41, 201, 421, 431, 111, 311) // indias atcally ward in tree construction: // indias atcally indias at tree is 0.2225 = 33 / 400 // plot(prune.dtas, propercise) // itle(sime-Frued tree over the training sample at size 6') // itle(sime-Frued tree over the training sample at size 6') // itle(sime-Frued tree over the tree is 0.6 // plot(prune.dtas, type="classified") // itle(sime-dtas) // i</pre>		
<pre>nub_tree(tree = tree.dat, nodes = e(t. 201, 421, 431, 111, 111) Variables actually used in tree construction: [11] "Shadwater of "Price" 'Advertising "CompTrice" Headmail mean deviance: 0.944 = 379.8 / 394 Hischassfitcation error trate: 0.225 = 33 / 400 &gt; bold (prume.data, prety-0) &gt; table(phit, data, prety-0, prety, prety,</pre>	Classification tree:	*
<pre>blid Variable saturally used in tree construction: [1] TshalveLoc " Fries" "Advertising "Comprise" Maker of Commission Decore (A = 175.8 f &gt; 194 Hischassification error rate: 0.2325 = 93 / 400 &gt; blit(prune.data,perty=0) &gt; blit(prune.data,tety=0) &gt; till(prune.data,type="class") &gt; t</pre>	<pre>snip.tree(tree = tree.data, nodes = c(4L, 20L, 42L, 43L, 11L,</pre>	
<pre>Variable actually used in tree construction: [11 *halvetus' 'Fries' 'Advettising' "Comprise" Rumber of terminal node: 6 Advettising' "Comprise" Rumber of terminal node: 6 Advettising' "Comprise" Rumber of terminal node: 6 Advettising' "Comprise" Pathod for trace (0.225 = 03 / 400 &gt; itilizionan-Adta) &gt; itilizionan-Adta &gt; itilizionan-Adta, proty-0) &gt; itilizionan-Adta, proty-0 &gt; itilizionan-Adta, proty-0 &gt; itilizionan-Adta, proty-0 &gt; itilizionan-Adta, proty-0 &gt; producta, eseq-pruse. (res (tres. data) &gt; prome. Adta. seq-pruse. (res (tres. data) &gt; producta, eseq-pruse. (res (tres. data)) &gt; pruse. (tres. data, proty-0, nored (tres.)) &gt; pruse. (tres. data, proty-0, nored (tres.)) &gt; pruse. (tres. data, best-6, newdata-data[-train,]) &gt; pruse. (tres. data, best-6, newdata-data[-train,]) &gt; pruse. (tres. data, best-6, newdata-data[-train,]) &gt; pruse. (tres. data, best-6, newdata-data[-train,], type="class") &gt; data. (set-data[-train,") &gt; this (prot, data. (set) data. (set-data[-train,") &gt; this (prot, data. (set) data. (set-data[-train,") &gt; this (prot, data. (set) proty (tres. data, best-6, newdata-data[-train,], type="class") &gt; this (prot, data. (set) data. (set-data[-train,") &gt; this (prot, data. (set) data. (set) prot, data. (set</pre>	3L))	
<pre>[1] "ShalveLos" "Frie" "Advertising "CompBrie" Number of terminal nodes: 0 Facifical seam deviance: 0.544 - 378.6 / 394 Final terminal nodes: 0.5255 - 83 / 400 Final terminal terminal nodes: 0.5255 - 83 / 400 Final terminal nodes: 0.525 - 83 / 400 Final terminal node: 0.555 - 83 / 400 Final terminal ter</pre>	Variables actually used in tree construction:	
<pre>hubber of terminal modes: 6 Hubber of terminal modes: 6 Heridania Remove deviance: 0.494 = 379.04 = 379.04 = 379.04 Hisclassification error rate: 0.2225 = 33 / 400 &gt; plot(prune.data) &gt; test(prune.data) &gt; test(prune.da</pre>	[1] "ShelveLoc" "Price" "Advertising" "CompPrice"	
<pre>heatdoal mean deviance: 0.944 = J79.4 / J94 HisclassFicture or rot set 0.225 = 93 / 400 &gt; plot(prume.data) &gt; bit(prume.data.pretty=0) &gt; tit(prume.data.pretty=0) &gt; tit(prume.data.</pre>	Number of terminal nodes: 6	Г
<pre>Hisciasification error rate: 0.225 = 03 / 400 &gt; plot(prune.data) &gt; plot(prune.data)</pre>	Residual mean deviance: 0.964 = 379.8 / 394	L.
<pre>&gt; plot(prume.data) &gt; text(prume.data) &gt; text(prume.data)preture() &gt; title(paine"Prumed tree over the tree size 6 &gt; yhat-predic(prume.data,type="class") &gt; table(paint,r)) yha &gt; for we he is is Yes 70:149 &gt; fshow the sequence of prumed trees &gt; attach(tree.data) &gt; prume.data.seq) Length Class Mode size 22 -none= numeric % mode in -none= -ohreater &gt; attach(tree.data) &gt; summeric % mode in -none= -ohreater &gt; attach(data) &gt; set.seq(1) &gt; trais phat.data.test) &gt; the is prive.data.test=forms.tree size 6 &gt; attach(data) &gt; set.seq(1) &gt; trais phat.data.test) &gt; table(tree.data), mrow(data)/2) &gt; prume.data.test=forms.tree.test=forms.tree.test=forms.tree.test=forms.tree &gt; attach(data) &gt; set.seq(1) &gt; trais phat.data.test) &gt; table(phat.data.test) &gt; table(phat.test=table(phat.test) &gt; table(phat.test) &gt; table(phat.test) &gt; table(phat.test) &gt; table(pha</pre>	Misclassification error rate: 0.2325 = 93 / 400	
<pre>&gt; tilt(spins.dts.proty-of) &gt; tilt(spins.dts.proty-of) &gt; tilt(spins.dts.regues the training sample at size 6') &gt; tabut the training sample at size 6' &gt; tabut the training sample at size 6' &gt; tabut the training sample at size 6') &gt; tabut the training sample at size 6' &gt; tabut the training sample at size 6' &gt; tabut the training sample at size 6' &gt; some typics.dts.seq </pre>	> plot(prune.data)	
<pre>&gt; titlemain="Frund tree over the training maple at mise 6") &gt; f Estimate the train-USE at tree mise 6 &gt; yht-predict(prume.data,type="class") &gt; table(pht,ty) y y stable(pht,ty) y stable(pht,data.test) lenght Class Mode size 22 -none-numeric data user-one-numeric 23 - none-numeric 24 - none-numeric 25 - none-numeric 25 - none-numeric 26 - none-numeric 27 - none-numeric 28 - none-numeric 29 - none-numeric 20 - none-numeric 21 - none-numeric 22 - none-numeric 23 - none-numeric 24 - none-numeric 25 - none-numeric 26 - none-numeric 27 - none-numeric 28 - none-numeric 29 - none-numeric 20 - none-numeric 21 - none-numeric 22 - none-numeric 23 - none-numeric 24 - none-numeric 25 - none-numeric 26 - none-numeric 27 - none-numeric 28 - none-numeric 28 - none-numeric 29 - none-numeric 20 - none-numer</pre>	> text(prune.data,pretty=0)	
<pre>p # Estimate the train-MSE at tree size 6 &gt; yhats-protectiverse.dsta.stype="class") &gt; table(phat.y) y yhat No Yes No 150 15 Yes 70 140 &gt; floot the negemond of prumed trees &gt; around dsta.secpryme.tree(tree.dsta) &gt; rummary(prume,dsta.sec) &gt; rummary(prume) &gt; rummary(prum) &gt; rummary(pru</pre>	> title(main="Pruned tree over the training sample at size 6")	
<pre>&gt; yhatperedict(prume.data,typ="class") &gt; table(pht.ty) y y that No Yes No 150 15 &gt; find the Segments of ground trees &gt; find the seqperume.tree(tree.data) &gt; prume.data.seq) Lenght Class Mode dis 22 -nonse numeric data cst-toms numeric k 22 -nonse numeric k 22 -n</pre>	> # Estimate the train-MSE at tree size 6	
<pre>&gt; tails (yhst,y) y phat No Yes No 158 15 Yes 78 143 &gt; # Show the sequence of prumed trees &gt; attach(tree.data) prume.data.seq prume.data.seq rome=rimeeric dev 22 - none=numeric dev 22 - none=numeric dev 22 - none=numeric ethod 1 - none=-ohreater &gt; # Estimate the test=HEM attree size 6 &gt; attach(data) &gt; # set.secl(1) &gt; train = sample(linrow(data), mrow(data)/2) &gt; prume.data.stemprome.tree.data,best=0,mendata=data[-train,]) &gt; ydatperedic(tree.data,best=0,mendata=data[-train,]) &gt; ydatperedic(tree.data,best=0,mendata=data[-train,]) &gt; tata.tot=data[-train,"] &gt; tata.tot=data[-train,"] &gt;</pre>	> vhat=predict(prune.data.tvpe="class")	
<pre>y = tri y = tri W = 79 149 &gt; # Show the sequence of pruned trees = # Show the sequence of pruned trees = # Show the sequence of tree.data &gt; prune.data.set.data &gt; = noone - numeric &amp; 22 - noone - numeric &amp; tata.data &gt; = noone - numeric &amp; trees the tast-MEX at tree size &gt; # Skinate the tast-MEX at tree size &gt; # Skinate the tast-MEX at tree size &gt; # Skinate the tast-MEX at tree size &gt; prune.data.test y data.test-data[-train,], type="class") &gt; table(phat.data.test) data.test y tast.set y at the top the form source file: mytree.R . *actree §y Skyars , model(tree) op_mys(WID) ov_tree // cross-validation with optimal tree size &gt; ************************************</pre>	> table(vhat.v)	
<pre>phat No Yes No 15 Yes 78 145 Yes 78 Y</pre>	y	
<pre>No 150 15 Yes 78 149 &gt; # Show the sequence of prund trees &gt; # stack(tree.data) &gt; prune.data.seq=prune.tree(tree.data) &gt; prune.data.seq=prune.tree(tree.data) &gt; prune.data.seq=prune.tree data.set data.set &gt; - come numeric &amp; 22 - none numeric &amp; 22 - none numeric &amp; 22 - none numeric &amp; 22 - none numeric &gt; # Satuants the test-JSE at tree size 6 &gt; # Satuants the test-JSE at tree size 6 # Satuants the test-JSE at the size 6 # Satuants the size for the size for the size 6 # Satuants the size for the size 6 # Satu</pre>	vhat No Yes	
<pre>Yes /9 t49 y # Show the sequence of pruned trees &gt; # Show the sequence of pruned trees &gt; # tach(tree.data) &gt; # summary (prune.data.seq) Lenght Class Mode size 22 - none- numeric withod 1 -none- numeric # Bruinate tast-MEM at tree size 6 &gt; # tach(data) # Harlingt tree.data, here-f, newdata-data[-train,]) &gt; # stel.sed() &gt; train = sample((inrow(data), mrow(data)/2) &gt; # prune.data.test-free.data, here-f, newdata-data[-train,]) &gt; yhat-predic(tree.data, here-f, newdata-data[-train,]) &gt; table(phat, data.test) + tactions the tree.tast is # the size for the set is # for the set is</pre>	No 158 15	
<pre>&gt; # Show the sequence of prund trees &gt; attach (ree.data) &gt; prune.data.seq=prune.tree(tree.data) &gt; prune.data.seq=prune.tree(tree.data) &gt; prune.data.seq=prune.data.seq leg 22 - nonse numeric dev 22 - nonse numeric dev 22 - nonse numeric method 1 - nonse - obsretie method 1 ([linror(data], mrov(data)/2]) &gt; stat.sed([linror(data], mrov(data)/2]) &gt; phal-predict(tree.data.best-fo,medata=data[-train,]) &gt; phal-predict(tree.data.best-fo,medata=data[-train,]) &gt; data.test-data.test-fo,medata=data[-train,],type="class") &gt; data.test-data.test-fo,medata=data[-train,],type="class") &gt; data.test-data.test-fo,medata=data[-train,],type="class") &gt; data.test-data.test-fo,medata=data[-train,],type="class") &gt; data.test-data.test-fo,medata=data[-train,],type="class") &gt; data.test-data.test-fo,medata=data[-train,],type="class") &gt; data.test-fo,medata=data[-train,],type="class") &gt; data.test-fo,medata=data[-train,],type="class") &gt; data.test-fo,medata=data[-train,],type="class") &gt; data.test-fo,medata=data[-train,],type="class"] &gt; data.test-fo,medata=data[-train,],type="</pre>	Yes 78 149	
<pre>&gt; stcsh(tree.data) &gt; prumo.data.exp;rumo.tree(res.data) &gt; prumo.data.exp;rumo.tree(res.data)) &gt; mumaria stcshort class Mode size 22 - none= numeria 4 2 - none= numeria 4 - num</pre>	> # Show the segmence of proped trees	
<pre>&gt; prom.data.seep=prome.tree(tree.data) &gt; numeric class.sed * conset.tage=prome.tage.tage=prome.tage.tage=prome tage=prome tage tage prome tage=prome tage tage prome tage prometation tage p</pre>	> a show one sequence or primer cross	
<pre>&gt; runsary Tyruse.dsta.seq) Length Class Mode size 22 -none-numeric dev 22 -none-numeric k 22 -none-numeric k 22 -none-numeric k - 22 -none-numeric k -</pre>	> wrong data segmentine type/type data)	
<pre>import Class Hode iis 22 -none-numeric der 22 -none-numeric der 22 -none-numeric etabol 1 -none-ohrreite bethod 1 -none-ohrreite etabol 1 -none-ohrreite set.seol(1) &gt; train = sample(linror(data), mrow(data)/2) &gt; train = sample(linror(data)/2) &gt; train = sample(linror(d</pre>	> prune vaca sequences teo (cree vaca)	
<pre>ite 22 - room= numeric (dv 22 - room= numeric) (dv 22 - noom= numeric) k 22 - noom=</pre>	source a so	
<pre>fer is income - immeric is iscome - immeric ethod 1come - character &gt; # Estimate the test-HES at tree size 6 &gt; attach(data) &gt; est.sed(1) &gt; train = sample(inrow(data), mrow(data)/2) &gt; prume.data.test-grows.terme.data.hest-d, mrow(data-data[-train,]) &gt; yhat-prodict(tree.data,best-d, mrow(data-data[-train,]) &gt; yhat-prodict(tree.data,best-d, mrow(data-data[-train,]) &gt; table(phat,data.est) &gt; table(pha</pre>	Lengui class noue	
<pre>prov 10 - roome-character sethod 1 - roome-character &gt; # Estimate the test-HSE at tree size 6 &gt; attach(dat) &gt; set.seed(1) &gt; etrain = magle(1:nrow(data), nrow(data)/2) &gt; purperpeticit(tree.data,beeted,newsdata=data(-train,)) &gt; purperpeticit(tree.data,beeted,newsdata=data(-train,)) &gt; purpeticit(tree.data,beeted,newsdata=data(-train,)) &gt; purpeticit(tree.data,beeted,newsdata=data(-train,)) &gt; table(pht_data.test=t) data.test=data(-train,'y') &gt; table(pht_data.test=t) data.test=therest, data.test &gt;&gt; End of R output from source file: mytree.R </pre>		
<pre>set.sed() set.sed() set.set() set.set.set.set.set.set.set.set.set.set.</pre>		
<pre>percond i = "none-character = iss 6 &gt; bitabi(data) &gt; sitabi(data) &gt; train = sage[(1:rrow(data), mrow(data)/2) &gt; prome_data.test=percent.test_newdata=data[-train,]) &gt; phatoperclucture.data.hest=of_newdata=data[-train,], type="class") &gt; data.test=data[-train,"y"] &gt; table(phat_data.test=t) data.test=data[-train,"y"] &gt; table(phat_data.test=t) data.test=data[-train,"y"] &gt; table(phat_data.test=t) data.test=theta[-train,"y"] &gt; table(phat_data.test=theta[-train,"y"] &gt; table(phat_data.test=theta[-train,"y"] &gt; table(phat_data.test=theta[-train,"y"] &gt; table(phat_data.test=theta[-train,"y"] &gt; table(phat_data.test=theta[-train,"y"] &gt; table(phat_data.test=theta[-train,"y"] &gt; table(phat_data.test=theta[-train,"y"] &gt; table(phat_data.test=theta[-train,</pre>		
<pre>petsend() &gt; etsiand(ats.up) Cell-Vask at tree size 0 &gt; vask test-data(-train,') &gt; vash test-data(-train,') &gt; table(vast.data.test) data.test-data(-train,') &gt; table(vast.data.test) data.test &gt; table(vast.data.test) data.test &gt; table(vast.data.test) data.test-data(-train,') &gt; table(vast.data.test) data.test &gt; table(vast.data.test) data.test &gt; table(vast.data.test) </pre>	method 1 -hone-character	
<pre>&gt; bets resent(1) &gt; rest: = sample(1:prov(data), mrow(data)/2) &gt; prume_data.test-prume.tree(tree.data.best-6,newdata=data(-train,)) &gt; yhat-prof(tree.data.best-6,newdata=data(-train,), type="class") &gt; data.test-data(-train, "y") &gt; behale(phat.data.test) data.test &gt; that No Yes No 107 6 Yes 9 70 &gt; &gt; End of R output from source file: mytree.R . testree §y Savars , model(tree) op_mys(MIR) ov_tree // cross-validation with optimal tree size .</pre>		
<pre>&gt; Finite = Lip = (Liprovidata) / nrow(data) /2) &gt; prime, data constructions (res. data, baset, providate and ta(-train, )) &gt; prime, data constructions (res. data, baset, providate and ta(-train, ), type="class") &gt; data.test-data(-train, 'p') data.test-data(-train, 'p') data(-train, 'p') data.test-data(-train, 'p') data(-train, 'p') data(</pre>	action (usea)	
<pre>&gt; train = subje(1:hrow(okis)/2) &gt; train = subje(1:hrow(okis)/2) &gt; train = subje(1:hrow(okis)/2) &gt; train = subje(1:hrow(okis)/2) &gt; train(train, hrow(okis)/2) &gt; train(train, hrow(okis)/2)</pre>	> Bec. Becu(1)	
<pre>p prume.nata. data (strong trees (trees.tre</pre>	<pre>&gt; train = sample(i:nrow(data), nrow(data)/2)</pre>	
<pre>&gt; yistoppoutoilite.ite.itat,hesve,hemailemaile(-train,),iype=class') &gt; table(phit-(train, 've)) table(phit-(train, 've)) table(phit-(train, 've)) table(phit-(train, 've)) table(phit-(train, 've)) * No 107 6 Yes 3 78 &gt;&gt; End of R output from source file: mytree.R . *sotree §y Savars , model(tree) op_mys(MIR) or_tree // cross-validation with optimal tree size * * * * * * * * * * * * * * * * * * *</pre>	> prune.cata.test=prune.tree(tree.cata.pest=6, newcata.cata[-train.j])	
<pre>&gt; auta.tosto-cuta[-traih, 'y'] &gt; table(phat.tosto-cuta[-traih, 'y'] auta.tost ) table(phat.tosto-cuta] ) table(phat.</pre>	> ynat=predict(tree.data.best=0,newdata=data[-train,],type="class")	
<pre>&gt; table(matc.icst)     data.icst)     data.icst     d</pre>	where the state is a state of the state o	
and.s.test yhat No Yos No 107 6 Yes 3 75 > End of R output from source file: mytree.R .*setree §y §zvars , model(tree) op_mys(MIN) ov_tree // cross-validation with optimal tree size	<ul> <li>caste grad, data, cest)</li> </ul>	
<pre>ymat no res No 107 e No 107 e Yes &gt; 76 '-&gt; End of R output from source file: mytrees.R *actree \$y \$xvars , model(tree) op_mys(M(H)) ov_tree // cross-validation with optimal tree size '</pre>	data.test	
No 107 6 Yes 9 78 > > End of R output from source file: mytree.R . *setree §y Szvars , model(tree) op_mys(MIN) ov_tree // cross-validation with optimal tree size	ynat No Yes	
<pre>Yes y 76 &gt;&gt; End of R output from source file: mytree.R . *sotree §y \$xvars , model(tree) op_mys(MIN) ov_tree // cross-validation with optimal tree size</pre>	No 107 6	
<pre>&gt; End of R output from source file: mytree.R . *sotree §y Savars , model(tree) op_mys(MIN) ov_tree // cross-validation with optimal tree size * * ********************************</pre>	Yes 9 78	
> End of R output from source file: mytree.R . *sotree §y §xvars , model(tree) op_mys(MIN) ov_tree // cross-validation with optimal tree size *	>	
. *sctree §y §xvars , model(tree) op_mys(MIN) ov_tree // cross-validation with optimal tree size *	> End of R output from source file: mytree.R	
	. testree Sv Svurre . model(tree) on eve(WTN) ov tree	=
	, serve y writes , moust (tree) of system, of tree // tross-variation with optimal tree size	*
	Г	

18



#### R output visible as Stata output - 3



#### Pruned tree over the training sample at size 6

19

#### **Optimal tree size** via cross-validation - 1

sctree \$y \$xvars , model(tree) op\_sys(WIN) cv\_tree

Regression Tree over the whole training sample



RSS sequence of pruned trees over the whole training sample



Jeviance



#### **Optimal tree size via cross-validation - 2**



CV-based optimal tree



21

## Thanks for your attention

